

PATENT
EMCR:054
EMC-99-176

APPLICATION FOR UNITED STATES LETTERS PATENT

for

Preparation of Metadata for Splicing of Encoded MPEG Video and Audio

by

John Forecast
Daniel Gardere
Peter Bixby
Sorin Faibish
Wayne W. Duso

0011220-90804560

EXPRESS MAIL MAILING LABEL	
NUMBER	<u>EL315197510 US</u>
DATE OF DEPOSIT	<u>03/31/00</u>
I hereby certify that this paper or fee is being deposited with the United States Postal Service "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service under 37 C.F.R. 1.10 on the date indicated above and is addressed to: Assistant Commissioner for Patents, Washington D.C. 20231.	
<u>Martha S. Rineda</u>	
Signature	

1 RELATED APPLICATIONS

2 This application is a division of Provisional Application Ser. No. 60/174,360 filed
3 Jan. 4, 2000, incorporated herein by reference.
4

5 BACKGROUND OF THE INVENTION

6 1. Field of the Invention.

7 The present invention relates to processing of compressed audio/visual data, and
8 more particularly to splicing of streams of audio/visual data.
9

10 2. Background Art.

11 It has become common practice to compress audio/visual data in order to reduce
12 the capacity and bandwidth requirements for storage and transmission. One of the most
13 popular audio/video compression techniques is MPEG. MPEG is an acronym for the
14 Moving Picture Experts Group, which was set up by the International Standards
15 Organization (ISO) to work on compression. MPEG provides a number of different
16 variations (MPEG-1, MPEG-2, etc.) to suit different bandwidth and quality constraints.
17 MPEG-2, for example, is especially suited to the storage and transmission of broadcast
18 quality television programs.

19 For the video data, MPEG provides a high degree of compression (up to 200:1) by
20 encoding 8 x 8 blocks of pixels into a set of discrete cosine transform (DCT)
21 coefficients, quantizing and encoding the coefficients, and using motion compensation
22 techniques to encode most video frames as predictions from or between other frames. In
23 particular, the encoded MPEG video stream is comprised of a series of groups of pictures
24 (GOPs), and each GOP begins with an independently encoded (intra) I frame and may

1 is given in "A guide to MPEG Fundamentals and Protocol Analysis (Including DVB and
2 ATSC)," Tektronix Inc., 1997, incorporated herein by reference.

3 Splicing of audio/visual programs is a common operation performed, for example,
4 whenever one encoded television program is switched to another. Splicing may be done
5 for commercial insertion, studio routing, camera switching, and program editing. The
6 splicing of MPEG encoded audio/visual streams, however, is considerably more difficult
7 than splicing of the uncompressed audio and video. The P and B frames cannot be
8 decoded without a preceding I frame, so that cutting into a stream after an I frame renders
9 the P and B frames meaningless. The P and B frames are considerably smaller than the I
10 frames, so that the frame boundaries are not evenly spaced and must be dynamically
11 synchronized between the two streams at the time of the splice. Moreover, because a
12 video decoder buffer is required to compensate for the uneven spacing of the frame
13 boundaries in the encoded streams, splicing may cause underflow or overflow of the
14 video decoder buffer.

15 The problems of splicing MPEG encoded audio/visual streams are addressed to
16 some extent in Appendix K, entitled "Splicing Transport Streams," to the MPEG-2
17 standard ISO/IEC 13818-1 1996. Appendix K recognizes that a splice can be "seamless"
18 when it does not result in a decoding discontinuity, or a splice can be "non-seamless"
19 when it results in a decoding discontinuity. In either case, however, it is possible that the
20 spliced stream will cause buffer overflow.

21 The Society of Motion Picture and Television Engineers (SMPTE) apparently
22 thought that the ISO MPEG-2 standard was inadequate with respect to splicing. They
23 promulgated their own SMPTE Standard 312M, entitled "Splice Points for MPEG-2

1 Transport Streams,” incorporated herein by reference. The SMPTE standard defines
2 constraints on the encoding of and syntax for MPEG-2 transport streams such that they
3 may be spliced without modifying the packetized elementary stream (PES) packet
4 payload. The SMPTE standard includes some constraints applicable to both seamless
5 and non-seamless splicing, and other constraints that are applicable only to seamless
6 splicing. For example, for seamless and non-seamless splicing, a splice occurs from an
7 Out Point on a first stream to an In Point on a second stream. The Out Point is
8 immediately after an I frame or P frame (in presentation order). The In Point is just
9 before a sequence header and I frame in a "closed" GOP (i.e., no prediction is allowed
10 back before the In Point).

11 As further discussed in Norm Hurst and Katie Cornog, “MPEG Splicing: A New
12 Standard for Television - SMPTE 312M,” SMPTE Journal, Nov. 1998, there are two
13 buffering constraints for seamless splicing. The startup delay at the In Point must be a
14 particular value, and the ending delay at the Out Point must be one frame less than that.
15 Also, the old stream must be constructed so that the video decoder buffer (V BV buffer)
16 would not overflow if the bit rate were suddenly increased to a maximum splice rate for a
17 period of a splice decoding delay before each Out Point.

18 SUMMARY OF THE INVENTION

19
20 In accordance with a first aspect, the invention provides a method of
21 preparing metadata for splicing of a transport stream. The transport stream includes
22 video access units encoding video presentation units representing video frames. The
23 video access units of the transport stream encode the video presentation units using a data

1 compression technique and contain a variable amount of compressed video data. The
2 method includes a file server ingesting the transport stream, and storing the transport
3 stream in a file in data storage. Concurrently with storing the transport stream in the file
4 in data storage, the file server computes metadata for splicing of the transport stream, and
5 stores the metadata for splicing in the file.

6 In accordance with another aspect, the invention provides a data storage device
7 containing a file of data of a transport stream including video access units encoding video
8 presentation units representing video frames. The video access units of the transport
9 stream encode the video presentation units using a data compression technique and
10 contain a variable amount of compressed video data. The file also contains an index to
11 groups of pictures (GOPs) in the transport stream. The index to the groups of pictures
12 includes pointers to transport stream file data of respective ones of the GOPs. The file
13 further contains attributes of the GOPs computed from the data of the transport stream.
14 The attributes of the GOPs are also indexed by the index to the groups of pictures.

15 16 BRIEF DESCRIPTION OF THE DRAWINGS

17 Other objects and advantages of the invention will become apparent upon reading
18 the following detailed description with reference to the accompanying drawings, in
19 which:

20 FIG. 1 is a block diagram of a video file server;

21 FIG. 2 is a perspective view showing the use of a set-top decoder box;

22 FIG. 3 is a block diagram showing a switch for splicing broadcast audio/visual
23 streams;

1 FIG. 4 is a block diagram of an MPEG decoder;

2 FIG. 5 is a diagram of the format of an MPEG transport packet stream;

3 FIG. 6 is a diagram of the format of an MPEG PES packet;

4 FIG. 7 is a diagram showing audio and video content in two MPEG transport
5 streams to be spliced;

6 FIG. 8 is a diagram showing aligned elementary video and audio streams resulting
7 from the splicing of the two MPEG transport streams in FIG. 7;

8 FIG. 9 is a diagram showing that audio access units are not aligned on audio PES
9 packet boundaries;

10 FIG. 10 is a logic table showing eight cases for the selection of audio presentation
11 units to be included in the splicing of two MPEG transport streams;

12 FIG. 11A is a diagram showing content of video and audio presentation unit
13 streams for the two MPEG transport streams for a first case in the logic table of FIG. 10;

14 FIG. 11B is a diagram showing the content of video and audio presentation unit
15 streams resulting from a first possible splicing of the two MPEG transport streams shown
16 in FIG. 11A;

17 FIG. 11C is a diagram showing the content of video and audio presentation unit
18 streams resulting from a second possible splicing of the two MPEG transport streams
19 shown in FIG. 11A;

20 FIG. 12A is a diagram showing content of video and audio presentation unit
21 streams for the two MPEG transport streams for a second case in the logic table of FIG.
22 10;

1 FIG. 17B is a diagram showing the content of video and audio presentation unit
2 streams resulting from a first possible splicing of the two MPEG transport streams shown
3 in FIG. 17A;

4 FIG. 17C is a diagram showing the content of video and audio presentation unit
5 streams resulting from a second possible splicing of the two MPEG transport streams
6 shown in FIG. 17A;

7 FIG. 18A is a diagram showing content of video and audio presentation unit
8 streams for the two MPEG transport streams for an eighth case in the logic table of FIG.
9 10;

10 FIG. 18B is a diagram showing the content of video and audio presentation unit
11 streams resulting from splicing of the two MPEG transport streams shown in FIG. 18A;

12 FIG. 19 is a flow chart of a procedure for splicing MPEG clips;

13 FIG. 20A is a graph of video buffer level versus time for decoding the end of a
14 first MPEG clip;

15 FIG. 20B is a graph of video buffer level versus time for decoding the beginning
16 of a second MPEG clip;

17 FIG. 21 is a graph of video buffer level versus time for decoding of a seamless
18 splicing of the first MPEG clip to the second MPEG clip;

19 FIG. 22 is a flow chart of a basic procedure for seamless splicing of video
20 streams;

21 FIG. 23 is a first portion of a flow chart of a procedure for splicing video streams;

22 FIG. 24 is a second portion of the flow chart begun in FIG. 23;

23 FIG. 25 is a first portion of a flow chart of a procedure for splicing audio streams;

1 stamps (PTS) and extrapolated program clock reference (PCR) time stamps for an audio
2 elementary stream;

3 FIG. 37 is a schematic diagram of circuitry for computing an expected maximum
4 and an expected minimum audio buffer level from the estimated average audio buffer
5 level and standard deviation of the average audio buffer level from the digital filter
6 circuitry in FIG. 36;

7 FIG. 38 is a flow chart of a procedure for computing an offset for the video
8 decode time stamps (DTS) of the second clip for splicing the second clip onto the first
9 clip;

10 FIG. 39 is a flow chart of a procedure for computing an offset for the audio
11 presentation time stamps (PTS) of the second clip for splicing the second clip onto the
12 first clip;

13 FIG. 40 is a flow chart of a procedure for computing an offset for the program
14 clock reference (PCR) time stamps of the second clip for splicing the second clip to the
15 first clip;

16 FIG. 41 is a flow chart of a procedure for re-stamping a second clip for splicing of
17 the second clip to the first clip;

18 FIG. 42 is a diagram of macroblocks in a video frame;

19 FIG. 43 is a diagram showing non-obsolete audio packets in a first TS stream
20 following the end of video at an Out Point and null packets and obsolete audio packets in
21 a second TS stream following the beginning of video at an In Point;

22 FIG. 44 is a flow chart of a re-formatting procedure that replaces the null packets
23 and obsolete audio packets in FIG. 43 with the non-obsolete audio packets in FIG. 43;

1 FIG. 45 is a diagram showing MPEG Transport Stream (TS) metadata
2 computation and storage of the metadata in the header of an MPEG TS data file;

3 FIG. 46 is a block diagram of the preferred format of a GOP index introduced in
4 FIG. 45;

5 FIG. 47 is a flow chart showing decimation of the GOP index;

6 FIG. 48 is a flow chart showing metadata computations for a next GOP in an
7 ingested TS;

8 FIG. 49 is a block diagram of various blocks in the stream server computer of the
9 video file server of FIG. 1 for computing MPEG metadata during ingestion of an MPEG
10 TS, and for performing real-time MPEG processing such as seamless splicing in real-time
11 during real-time transmission of a spliced MPEG TS;

12 FIG. 50 is a diagram showing flow of control during a metered file transfer using
13 the video server of FIG. 1;

14 FIG. 51 is a block diagram of play lists in the video file server of FIG. 1, showing
15 that a stream server play list is maintained as a window into a control station play list;
16 and

17 FIG. 52 is a flow chart showing the use of seamless splicing for repair of a
18 temporarily corrupted TS.

19 While the invention is susceptible to various modifications and alternative forms,
20 specific embodiments thereof have been shown in the drawings and will be described in
21 detail. It should be understood, however, that it is not intended to limit the form of the
22 invention to the particular forms shown, but on the contrary, the intention is to cover all

1 modifications, equivalents, and alternatives falling within the scope of the invention as
2 defined by the appended claims.

3

4 DESCRIPTION OF ILLUSTRATIVE EMBODIMENTS

5 Turning now to FIG. 1 of the drawings, there is shown a video file server
6 generally designated 20 which may use the present invention. The video file server 20
7 includes an array of stream servers 21, at least one control server 28, 29, a cached disk
8 array storage subsystem 23, and an optional tape silo 24. The video file server 20 is a
9 high performance, high capacity, and high-availability network-attached data server. It
10 provides the ability for multiple file systems to exist concurrently over multiple
11 communication stacks, with shared data access. It also allows multiple physical file
12 systems to co-exist, each optimized to the needs of a particular data service.

13 The video file server 20 is managed as a dedicated network appliance, integrated
14 with popular network operating systems in a way, which, other than its superior
15 performance, is transparent to the end user. It provides specialized support for real-time
16 data streams used in live, as well as store-and-forward, audio-visual applications.
17 Therefore, the video file server 20 is suitable for a wide variety of applications such as
18 image repositories, video on demand, and networked video applications, in addition to
19 high-end file server applications such as the Network File System (NFS, version 2 and
20 version 3) (and/or other access protocols), network or on-line backup, fast download, etc.

21 The clustering of the stream servers 21 as a front end to the cached disk array 23
22 provides parallelism and scalability. The clustering of random-access memory in the
23 stream servers 21 provides a large capacity cache memory for video applications.

1 Each of the stream servers 21 is a high-end commodity computer, providing the
2 highest performance appropriate for a stream server at the lowest cost. The stream
3 servers 21 are mounted in a standard 19" wide rack. Each of the stream servers 21, for
4 example, includes an Intel processor connected to an EISA or PCI bus and at least 64
5 MB of random-access memory. The number of the stream servers 21, their processor
6 class (i486, Pentium, etc.) and the amount of random-access memory in each of the
7 stream servers, are selected for desired performance and capacity characteristics, such as
8 the number of concurrent users to be serviced, the number of independent multi-media
9 programs to be accessed concurrently, and the desired latency of access to the multi-
10 media programs.

11 Each of the stream servers 21 contains one or more high-performance FWD (fast,
12 wide, differential) SCSI connections to the back-end storage array. Each of the stream
13 servers 21 may also contain one or more SCSI connections to the optional tape silo 24.
14 Each of the stream servers 21 also contains one or more outbound network attachments
15 configured on the stream server's EISA or PCI bus. The outbound network attachments,
16 for example, are Ethernet, FDDI, ATM, DS1, DS3, or channelized T3 attachments to data
17 links to a network 25. Each of the stream servers 21 also includes an additional Ethernet
18 connection to a dual redundant internal Ethernet link 26 for coordination of the stream
19 servers with each other and with one or more controller servers 28, 29.

20 The controller servers 28, 29 are dual redundant computers 28, 29, each of which
21 is similar to each of the stream servers 21. Each of the dual redundant controller servers
22 28, 29 has a network attachment to a bidirectional link 30 in the network 25, through
23 which each of the controller servers 28, 29 can conduct service protocols. The service

1 protocols include one or more standard management and control protocols such as the
2 Simple Network Management Protocol (SNMP), and at least one Continuous Media File
3 Access Protocol supporting real-time multi-media data transmission from the stream
4 servers 21 to the network 25.

5 Each of the dual redundant controller servers 28, 29 has an Ethernet connection to
6 the local Ethernet link 26. Each of the controller servers 28, 29 also has a connection to a
7 serial link 31 to a media server display and keyboard 32. The controller servers 28, 29
8 run a conventional operating system (such as Windows NT or UNIX) to provide a hot-
9 failover redundant configuration. An active one of the dual redundant controller servers
10 28, 29 functions as a media server controller for the video file server 20. The active one
11 of the controller servers 28, 29 also allows management and control of the server
12 resources from the network using standard protocols, such as the Simple Network
13 Management Protocol (SNMP). The active one of the controller servers 28, 29 may also
14 provide lock management if lock management is not provided by the cached disk array
15 23.

16 For multi-media data transfer, the active one of the controller servers 28, 29
17 assigns one of the stream servers 21 to the network client 54 requesting multi-media
18 service. The network 25, for example, has conventional transmission components 53
19 such as routers or ATM switches that permit any one of the clients 54 to communicate
20 with any one of the stream servers 21. The active one of the controller servers 28, 29
21 could assign a stream server to a network client by a protocol sending to the client the
22 network address of the stream server assigned to send or receive data to or from the
23 client. Alternatively, the active one of the controller servers 28, 29 could communicate

1 with a router or switch in the transmission components 53 to establish a data link between
2 the client and the stream server assigned to the client.

3 The cached disk array 23 is configured for an open systems network environment.
4 The cached disk array 23 includes a large capacity semiconductor cache memory 41 and
5 SCSI adapters 45 providing one or more FWD SCSI links to each of the stream servers
6 21 and to each of the dual redundant controller servers 28, 29. The disk array 47 may
7 store data using mirroring or other RAID (redundant array of inexpensive disks)
8 techniques to recover from single disk failure. Although simple mirroring requires more
9 storage disks than the more complex RAID techniques, it has been found very useful for
10 increasing read access bandwidth by a factor of two by simultaneously accessing each of
11 two mirrored copies of a video data set. Preferably, the cached disk array 23 is a
12 Symmetrix 5500 (Trademark) cached disk array manufactured by EMC Corporation, 171
13 South Street, Hopkinton, Mass., 01748-9103.

14 The tape silo 24 includes an array of SCSI adapters 50 and an array of read/write
15 stations 51. Each of the read/write stations 51 is connected via a respective one of the
16 SCSI adapters 50 and a FWD SCSI link to a respective one of the stream servers 21 or
17 each of the redundant controller servers 28, 29. The read/write stations 51 are controlled
18 robotically in response to commands from the active one of the controller servers 28, 29
19 for tape transport functions, and preferably also for mounting and unmounting of tape
20 cartridges into the read/write stations from storage bins.

21 Further details regarding the structure and operation of the video file server 20 are
22 found in Wayne Duso and John Forecast, "System Having Client Sending Edit
23 Commands to Server During Transmission of Continuous Media from One Clip in Play

1 List for Editing the Play List," U.S. Patent 5,892,915, issued April 6, 1999, incorporated
2 herein by reference. For practicing the present invention, the tape library 52 or cached
3 disk array 47 stores video clips in a compressed format. Each clip, for example, is a
4 recorded MPEG transport stream, including a video elementary stream and one or more
5 audio elementary streams synchronized to the video elementary stream. By using the
6 splicing techniques as described below, it is possible for the video file server to make a
7 seamless transition to a second clip from an intermediate location in a first clip during
8 real-time audio/video data transmission from the video file server 20 to one of the clients
9 54. In this regard, for the purposes of interpreting the appended claims, "seamless
10 splicing" should be understood to mean a process that will produce a spliced transport
11 stream, the play-out of which is substantially free from any audio-visual artifact that the
12 human auditory and visual system can detect.

13 With reference to FIG. 2, there is shown another application for seamless splicing
14 of MPEG transport streams. In this application, a set-top decoder box 61 receives a
15 number of MPEG transport streams from a coaxial cable 62. Each of the MPEG
16 transport streams encodes audio and video information for a respective television
17 channel. A viewer (not shown) may operate a remote control 63 to select one of the
18 channels for viewing on a television 64. The decoder box 61 selects the MPEG transport
19 stream for the desired channel and decodes the transport stream to provide a conventional
20 audio/visual signal (such as an NTSC composite analog audio/video signal) to the
21 television set.

22 In the set-top application as shown in FIG. 2, a problem arises when the viewer
23 rapidly scans through the channels available from the decoder 61. If a simple

1 includes digital data specifying the color and intensity of each pixel in a video frame. A
2 first audio decoder 95 receives A-PES packets from the first audio buffer 92 and
3 produces audio presentation units (APUs) for a first audio channel. An audio
4 presentation unit, for example, includes digital data specifying a series of audio samples
5 over an interval of time. A second audio decoder 96 receives A-PES packets from the
6 second audio buffer 93 and produces APUs for a second audio channel. The first and
7 second channels, for example, are right and left stereo audio channels.

8 For seamless splicing of MPEG transport streams, it is not necessary to decode
9 the video and audio elementary streams down to the presentation unit level, nor is it
10 necessary to simulate the video and audio buffers. Instead, the transport stream need only
11 be parsed down to the level of the packetized elementary streams and access units, and
12 the video and audio buffers need be considered only to the extent of avoiding buffer
13 overflow or underflow. As will be described below, buffer overflow or underflow can be
14 avoided by estimating buffer level based on program clock reference (PCR) and decode
15 time stamp (DTS) values. Seamless splicing can be done independently of the method of
16 audio encoding, although the estimation of buffer level can be made more precise by
17 taking into consideration certain encoded data statistics, which happen to be dependent
18 on the type of audio encoding. It is desired to provide a generic splicing method in which
19 no constraining assumptions are made about various encoding parameters such as frame
20 rate, audio bit rate, and audio sampling frequency. It is also desired to achieve splicing
21 directly on the transport streams with as little complexity as possible.

22 FIG. 5 is a diagram showing the syntax of the MPEG-2 Transport Stream. This
23 diagram is a relevant portion of Figure F.1 of Annex F of the MPEG-2 standards

1 document ISO/IEC 13818-1. The MPEG-2 Transport Stream is comprised of a series of
2 188 byte TS packets, each of which may include video, audio, or control information.
3 Seamless splicing, as described below, may involve modification of the payload unit start
4 indicator, the packet identifier (PID), the continuity counter field, the adaptation field
5 length in the adaptation field, and the program counter (PCR) time stamp again provided
6 in the adaptation field. If the data of a video PES packet or audio PES packet starts in the
7 payload of a TS packet, then the payload unit start indicator bit is set to a one.
8 Otherwise, if the TS packet contains the continuation of an already initiated audio or
9 video PES packet, then the payload unit start indicator bit is set to zero. Very typically
10 the payload unit start indicator will be changed by setting it to one at the first TS packet
11 of the audio for the second stream in the spliced Transport Stream. The original
12 continuity counter values of the second stream are modified so that the continuity counter
13 values in the spliced TS have consecutive values. The adaptation field length in the
14 adaptation fields of the last audio TS packet in the first stream and also the first audio TS
15 packet in the second stream within the spliced TS will typically need to be modified
16 during splicing in order to insert some stuffing bytes to generate full 188 byte sized valid
17 transport packets. The original PCR values from the second stream are uniformly
18 incremented in the spliced TS.

19 FIG. 6 is a diagram showing the syntax of an MPEG-2 PES packet. This diagram
20 is a relevant portion of Figure F.2 of Annex F of the MPEG-2 standards document
21 ISO/IEC 13818-1. The MPEG-2 PES packet may include video, audio, or control
22 information. Seamless splicing, as described below, may involve modification of the
23 PES packet length, and the data alignment indicator and presentation time stamp (PTS)

1 streams are parsed prior to splicing to determine the relative presentation time of the
2 video and audio information around the desired beginning and end points. In addition,
3 splicing is more difficult than just removing certain Transport Stream packets from the
4 first and second Transport Streams and concatenating the two streams. In general, the
5 audio data to keep and the audio data to discard will not be segregated into contiguous
6 blocks in the Transport Streams. Typically the splicing operation will involve re-
7 formatting of the audio data in the spliced Transport Stream, as discussed below with
8 reference to FIG. 43.

9 As shown in FIG. 8, the portion of the first Transport Stream prior to the end
10 point has been parsed into a video PES stream 111 and an audio PES stream 112, and the
11 portion of the second Transport Stream after the beginning point has been parsed into a
12 video PES stream 113 and an aligned audio PES stream 114. The two video PES streams
13 111, 113 have been jointed together at a dashed cut line 115, and the two audio PES
14 streams have been also jointed at the dashed cut line 115. The natural cut point for the
15 audio stream, however, is not between video PES boundaries, and instead it is between
16 audio access units (AAU) which are decoded to produce corresponding audio
17 presentation units (APU). Therefore, there may be a slight gap or overlap at the cut line
18 115 between the AAUs from the first Transport Stream and the AAUs from the second
19 Transport Stream. The gap or the overlap is removed during a reformatting operation in
20 which the spliced Transport Stream is produced from the parsed video PES stream and
21 the parsed audio PES stream. Typically the reformatting operation will slightly shift the
22 alignment of the audio presentation units from the second Transport Stream with respect
23 to their corresponding video presentation units.

1 Point (I frame of a closed GOP) in the second TS. For splicing, the time base of the
2 second TS is shifted to achieve video presentation continuity.

3 Because the AAUs are usually not aligned with the VPUs, there is an issue with
4 respect to the selection of AAUs to be included in the spliced TS. In general, audio
5 truncation (i.e., positioning of the cut with respect to the stream of AAUs in the first and
6 second TS) should always be done at the AAU boundaries. Fractional AAUs are useless
7 because the audio encoding algorithm is such that only whole AAUs can be decoded.
8 Audio truncation for the ending stream should be done with respect to the end of its last
9 VPU's presentation interval. Audio truncation for the beginning stream should be done
10 relative to the beginning of its first VPU's presentation interval. These general rules,
11 however, are insufficient to precisely specify which AAUs should be selected near the cut
12 for inclusion in the spliced TS.

13 A more precise set of rules for selection of AAUs near the cut takes into
14 consideration the concept of the "best aligned APU" and also takes into consideration the
15 audio buffer level that would be expected in the beginning (i.e., second) stream absent
16 splicing. The "best aligned final APU" of the ending (i.e., first) stream is defined as the
17 APU whose presentation interval ends within one APU interval centered about the time
18 of the cut. The "best aligned initial APU" of the beginning (i.e., second) stream is
19 defined as the APU whose presentation interval starts within one APU interval centered
20 about the time of the cut. As shown in the logic table of FIG. 10, there are eight possible
21 cases that can be identified in terms of the "best aligned final APU," the "best aligned
22 initial APU," and the presence of an audio gap or an audio overlap with respect to these

1 The decoding logic of FIG. 10 is acceptable when the expected mean audio buffer
2 level would be neither high nor low in the second stream absent splicing (i.e., in the
3 original form of the second stream). When such a mean audio buffer level would be high
4 or low for the second stream, additional solutions may be appropriate, as will be
5 described below with reference to FIGS. 27 to 35.

6 Except for the cases in FIGS. 11A and 17A, splicing involves truncating the first
7 audio stream at the end of the best aligned final APU, and starting the second audio
8 stream at the best aligned initial APU. The presentation time stamps of the best aligned
9 initial APU and all following APUs from the second stream are re-stamped so that they
10 follow next in sequence after the best aligned final APU. Since presentation time stamps
11 are not provided for each AAU but rather specified in the header field of audio PES
12 packets for the first AAU commencing in the payload of the PES packet, the above
13 mentioned re-stamping is achieved by modifying only these specified presentation time
14 stamps. Further processing is required at the elementary stream level for modifying the
15 audio PES packet carrying the best aligned final APU, and modifying the audio PES
16 packet carrying the best aligned initial APU. The audio PES packet carrying the best
17 aligned final APU is modified by truncation of AAU data after the AAU associated with
18 the best aligned final APU, and modifying the PES packet size (in the corresponding PES
19 packet header field) accordingly. The audio PES packet carrying the best aligned initial
20 APU is modified by deleting the AAU data preceding the AAU associated with the best
21 aligned initial APU, and modifying the PES packet size (in the corresponding PES packet
22 header field) accordingly. In addition and as mentioned above, the audio PES packet
23 carrying the best aligned initial APU and all subsequent audio PES packets are modified

1 by re-stamping their PTS values to follow in sequence from the PTS value of the audio
2 PES packet carrying the best aligned final APU. The cases in FIGS. 11A and 17A
3 involve similar truncation and modification operations, but in these cases either an
4 additional APU is included in between the best aligned APUs (case of FIG. 11A) or one
5 of the best aligned APUs is omitted (case of FIG. 17A). For the eight cases of audio
6 splicing identified in FIG. 10, it is possible to construct a spliced audio elementary stream
7 with no holes and no audio PTS discontinuity. As a consequence, an audio/video skew in
8 presentation time of magnitude at most half of an APU duration will be introduced
9 following the cut point in the spliced stream. This audio splicing technique can be
10 repeated any number of times with neither a failure to meet its structural assumptions nor
11 a degradation in this audio/video skew performance. The A/V skews introduced by the
12 multiple splices do not accumulate. Irrespective of the number of consecutive splices, the
13 worst audio/video skew at any point in time will be half of the APU duration. At each
14 splice point, at the termination of the APUs and VPU of the first stream, the total audio
15 and video presentation durations up to that point will be almost matching each other, i.e.,
16 $|\text{video_duration} - \text{audio_duration}| \leq (1/2) \text{APU_duration}$. Therefore always the proper
17 amount of audio data will be provided by the audio splicing procedure described above.
18 The resulting audio stream is error-free and MPEG-2 compliant.

19 The audio and video elementary streams must be recombined around and
20 following the splice point. This is conveniently done by reformatting of spliced
21 Transport Stream around and following the splice point. The truncation of the final PES
22 packet of the first audio stream will typically necessitate the insertion of some adaptation
23 field padding into its last transport packet. The deletion of some AAU data from the

1 beginning of the second audio stream's initial PES packet will typically necessitate the
2 editing of at most two audio transport packets.

3 In any MPEG-2 Transport Stream, the audio bit rate, over the span of a few VAU
4 durations, is substantially constant. The VAUs, however, are of varying sizes. Therefore
5 the relative positions of VAUs and AAUs associated with VPU and APU almost
6 aligned in time cannot be maintained constant. Almost always it is the case that the
7 AAUs are significantly delayed with respect to the corresponding VAUs for which the
8 decoded representations are almost synchronous. Therefore, splicing to achieve the
9 solutions for the cases of FIGS. 11A to 18A also involves transport packet buffering and
10 re-multiplexing. The delayed audio packets near the Out Point in the first TS stream are
11 temporarily stored in a buffer when the first TS stream is truncated based on the VAU of
12 the Out Point. Also, the spliced TS is reformatted by deletion of some obsolete audio
13 packets at the beginning of the second stream around the In Point, and repositioning of
14 some audio packets of the first stream just following the Out Point into the spliced TS.

15 With reference to FIG. 19, there is shown a top-level flow chart of the preferred
16 procedure for splicing MPEG Transport Streams. At least the portions of a first and
17 second MPEG TS stream around the Out Point and In Point, respectively, are assumed to
18 be stored in a buffer. The stored MPEG TS data for the first stream will be referred to as
19 a first clip, and the stored MPEG TS data for the second stream will be referred to as a
20 second clip.

21 In a first step 121, the splicing procedure receives an indication of a desired end
22 frame of the first clip and a desired start frame of the second clip. Next, in step 122, the
23 splicing procedure finds the closest I frame preceding the desired start frame to be the In

1 Point for splicing. In step 123, a video splicing subroutine is invoked, as further
2 described below with reference to FIGS. 23 to 24. In step 124, an audio splicing
3 subroutine is invoked, as further described below with reference to FIGS. 25 to 26.
4 Finally, in step 125, the concatenation of the first clip up to about the Out Point and the
5 second clip subsequent to about the In Point is re-formatted, including re-stamping of the
6 PTS and PCR values for the audio and video.

7 Considering now video splicing, the splicing procedure should ensure the absence
8 of objectionable video artifacts, preserve the duration of the spliced stream, and if
9 possible, keep all of the desired frames in the spliced stream. The duration of the spliced
10 stream should be preserved in order to prevent any time drift in the scheduled play-list.
11 In some cases, it is not possible to keep all of the original video frames due to buffer
12 problems. In such a case, one or more frames of the clip are replaced by frozen frames,
13 and this frame replacement is made as invisible as possible.

14 Management of the video buffer is an important consideration in ensuring the
15 absence of objectionable video artifacts. In a constant bit rate (CBR) and uniform picture
16 quality sequence, subsequent pictures typically have coded representations of drastically
17 different sizes. The encoder must manage the decoder's buffer within several constraints.
18 The buffer should be assumed to have a certain size defined in the MPEG-2 standard.
19 The decoder buffer should neither overflow nor underflow. Furthermore, the decoder
20 cannot decode a picture before it receives it in full (i.e. completely). Moreover, the
21 decoder should not be made to "wait" for the next picture to decode; this means that
22 every 40 ms in PAL and 1/29.97 second in NTSC, the decoder must have access to a full
23 picture ready to be decoded.

1 The MPEG encoder manages the video decoder buffer through decode time
2 stamps (DTS), presentation time stamps (PTS), and program clock reference (PCR)
3 values. With reference to FIG. 20A, for example, there is shown the video buffer level
4 during the playing of a first clip. The x-axis represents the time axis. The video buffer
5 level initially increases in a linear fashion over a segment 131 as the buffer is loaded at a
6 constant bit rate. Then over a time span 132, video data is displayed at frame intervals,
7 and the buffer is replenished at least to some extent between the frame intervals. At a
8 time T_e , the last video frame's data is finished being loaded into the video buffer. Then
9 the video buffer is periodically depleted to some extent at each subsequent video frame
10 interval, and becomes emptied at a time DTS_{L1} .

11 FIG. 20B shows the video buffer level for a second clip. The video buffer begins
12 to receive video data for the second clip at a time PCR_{e2} . (PCR_{e2} is extrapolated from the
13 value of the most recent received genuine PCR record, to the first byte of the picture
14 header sync word of the first video frame in the clip to start. The extrapolation adjusts
15 this most recently received genuine PCR record value by the quotient of the displacement
16 in data bits of the clip from the position where it appears in the second clip to the position
17 at which video data of the first frame of the second clip begins, divided by the data
18 transmission bit rate for transmission of the clip to the decoder.) The video buffer level
19 initially increases in a linear fashion over a segment 134 as the buffer is loaded at a
20 constant bit rate. However, the slope of the segment 134 in FIG. 20B may be
21 substantially different from the slope of the segment 131 in FIG. 20A. In each case, the
22 slope of the segment is proportional to the bit rate at which the data is loaded into the
23 video buffer. As shown, the video data of the second clip is received at the video buffer

clip, but also at a subsequent location of the second clip. If the second clip is encoded by an MPEG-2 compliant encoder, then video buffer underflow or buffer overflow will not occur at any time during the decoding of the clip. However, this guarantee is no longer valid if the DTS_{F2} - PCR_{e2} relationship at the beginning of the second clip is altered. Consequently, to avoid buffer problems, the buffer occupancy at the end of the first clip must be modified in some fashion. This problem is inevitable when splicing between clips having significantly different ending and starting buffer levels. This is why SMPTE has defined some splice types corresponding to well-defined buffer levels.

In order to seamlessly splice the first clip of FIG. 20A to the second clip of FIG. 20B, the content of the first clip (towards its end) is modified so that PCR_{e2} can just follow T_e (by one byte transmission time) and DTS_{F2} can just follow DTS_{L1} (by one video frame presentation interval). FIG. 21 shows the video buffer level for the splicing of the first clip to the second clip in this fashion. The content around the end of the first clip has been modified to provide a buffer emptying characteristic shown in dashed lines, such as the line segments 136, so that the buffer is emptied sooner of video data from the first clip. In particular, this is done by replacing a frame loaded into the video buffer over an interval 137 with a "freeze frame" having a selected amount of video data. The position of DTS_{L1} has not changed, the position of DTS_{F2} is one video frame interval after DTS_{L1} , and the relationship DTS_{F2} - PCR_{e2} is unchanged, but the position of T_e has been moved to T_e' in order to achieve the desired conditions for seamless video splicing.

FIG. 22 shows a flow chart of a seamless video splicing procedure that obtains the desired conditions just described above. In a first step 141, the first DTS of the second clip is anchored at one frame interval later than the last DTS of the first clip in order to

1 prevent a video decoding discontinuity. Then, in step 142, the procedure branches
2 depending on whether the PCR extrapolated to the beginning frame of the second clip
3 falls just after the ending time of the first clip. If so, then the splice will be seamless with
4 respect to its video content. Otherwise, the procedure branches to step 143. In step 143,
5 the content of the first clip is adjusted so that the PCR extrapolated to the beginning
6 frame of the second clip falls just after the ending time of the first clip. Therefore the
7 desired conditions for seamless video splicing are achieved.

8 With reference to FIG. 23, there is shown a more detailed flow chart of a seamless
9 video splicing procedure. In a first step 151, the procedure inspects the content of the
10 first clip to determine the last DTS/PTS of the first clip. This last DTS/PTS of the first
11 clip is designated DTS_{L1} . Next, in step 152, the procedure inspects the content of the first
12 clip to determine the time of arrival (T_e) of the last byte of the first clip. In step 153, the
13 procedure adds one frame interval to DTS_{L1} to find the desired first DTS location for the
14 second clip. The sum, designated DTS_{F1} , is equal to $DTS_{L1} + 1/FR$, where FR is the video
15 frame rate. In step 154, while keeping the DTS-PCR_e relationship unaltered, the
16 procedure finds the time instant, designated T_s , at which the first byte of the second clip
17 should arrive. This is done by calculating $T_{START} = DTS_{F2} - PCR_{e2}$, and $T_s = DTS_{F1} - T_{START}$.

18 Continuing in FIG. 24, in step 155, execution branches depending on whether T_s
19 is equal to T_e plus 8 divided by the bit rate. If not, then the clips to be spliced need
20 modification before concatenation, and execution branches to step 156. In step 156,
21 execution branches depending on whether T_s is less than T_e plus 8 divided by the bit rate.
22 If not, then there is an undesired gap in between the clips to be spliced, and execution
23 branches to step 157. In step 157, null packets are inserted into the clips to be spliced to

1 modified by dropping only one of them in either of the clips to reduce the mean audio
2 buffer level for the second clip. In step 173, if the mean audio buffer level does not
3 exceed the high threshold B, then execution continues to step 175. In step 175, the mean
4 audio buffer level for the second clip, assuming no modification made for splicing, is
5 compared to a low threshold, designated A. (A, for example, has a value of 33% of the
6 audio buffer capacity.) If this mean audio buffer level is less than the low threshold A,
7 then the procedure branches to step 176. In step 176, if the above-defined best aligned
8 AAUs do not achieve a forward skew, then the best aligned AAUs are modified by
9 appending only one extra AAU either after the best aligned AAU in the first clip or
10 before the best aligned AAU in the second clip to increase the mean audio buffer level for
11 the second clip.

12 In general, a forward skew of the AAUs from the second stream by incrementing
13 their presentation time instants tends to increase the mean audio buffer level. Therefore,
14 a forward skew is good if the mean audio buffer level is low for the second stream. A
15 backward skew of the AAUs from the second stream by decrementing their presentation
16 time instants tends to decrease the audio buffer level. Therefore, a backward skew is
17 good if the mean audio buffer level is high for the second stream.

18 In step 175, if the mean audio buffer level is not less than the low threshold A,
19 then the procedure continues to step 177 in FIG. 26. The procedure continues to step 177
20 also after steps 174 and 176. In step 177, the procedure removes all AAUs in the first
21 clip after the best aligned AAU in the first clip, and adjusts the last audio PES packet
22 header in the first clip to reflect the change in its size in bytes after the removal. In FIG.
23 26, step 178, the procedure finds the audio PES packet in the second clip which includes

1 the best aligned AAU in the second clip, and removes all AAUs preceding the best
 2 aligned one in this PES packet. Then in step 179, the procedure produces a PES packet
 3 header to encapsulate the best aligned AAU and the AAUs after it, and writes the PES
 4 packet size into the header. Finally, in step 180, the procedure calculates the required
 5 audio PTS offset (A_{offset}) to be used for re-stamping the audio of the second clip.

6 The preferred implementation of the audio splicing routine in FIGS. 26 and 27
 7 uses the logic shown in FIG. 27. Depending on whether the mean audio buffer level for
 8 the second clip, assuming no modifications are made for splicing, is greater than the high
 9 threshold B or less than the low threshold A, the eight cases of FIG. 10 are expanded to
 10 sixteen cases. The preferred solutions for these eight additional cases are shown in FIGS.
 11 28 to 35. When the mean audio buffer level for the second clip, assuming no
 12 modifications are made for splicing, is neither greater than the high threshold B nor less
 13 than the low threshold A, then the solutions shown in FIGS. 11 to 18 are immediately
 14 applicable.

15 A preferred method of estimating the mean audio buffer level of a clip is to use
 16 the product $(PTS_i - PCR_{ei})(\text{BIT RATE})$ as an indication of the audio buffer level. PTS_i
 17 denotes the i th audio PTS time stamp, and PCR_{ei} denotes the PCR value extrapolated to
 18 the bit position of PTS_i . Because the product $(PTS_i - PCR_{ei})(\text{BIT RATE})$ will fluctuate
 19 more rapidly than the mean audio buffer level, the computed values may be processed by
 20 a simple digital filter routine to obtain an estimated value of the mean audio buffer level
 21 at any point of a clip. Shown in FIG. 36, for example, is a digital filter schematic that
 22 includes a single first-order recursive stage 191 for computing an estimate of the mean
 23 audio buffer level ABV. The computation includes a scaling of $(PTS_i - PCR_{ei})(\text{BIT$

1 RATE) by a factor of $1/n_{av}$, where n_{av} is the effective number of samples over which the
2 mean is estimated. The scaled value is added to the previous estimate of the mean value
3 of ABV scaled by a “forgetting factor” of $1-1/n_{av}$. The previous value is stored in a
4 register 192. In a similar fashion, an estimate of the variance of the audio buffer level at
5 any point of a clip is computed by similar circuitry or computations depicted in FIG. 36.
6 For example, the estimate of the variance can be computed by a subtractor 193 that
7 calculates the deviation of each sample of $(PTS_i - PCR_{ei})(BIT\ RATE)$ from the estimated
8 mean audio buffer level, a squaring unit 194, and another first-order recursive filter stage
9 generally designated 195.

10 Instead of determining whether the mean audio buffer level is relatively high or
11 low for a clip, a determination can be made as to whether the audio buffer full level (i.e.,
12 audio buffer size) is within a certain number of estimated standard deviations from the
13 estimated mean audio buffer level, or whether the audio buffer empty level (e.g., zero
14 bytes) is within a certain number of estimated standard deviations from the estimated
15 mean audio level. In this case, the certain number can be selected based on the usual
16 statistics of the type of audio encoding that is employed, in order to ensure the absence of
17 audio buffer underflow or overflow within a desired level of confidence. In order to
18 make the comparisons very simple at the time of splicing, the maximum and minimum
19 expected deviations from the estimated average can be computed in advance for each
20 clip. For example, FIG. 37 shows in schematic form the computations necessary to
21 compute the maximum of the estimated mean buffer level AVB plus twice the estimated
22 standard deviation, and to compute the minimum of the estimated mean buffer level AVB
23 minus twice the standard deviation. The box 198, for example, outputs a binary value

1 indicating whether or not the input A is greater than the input B. The symbol 199 denotes
2 a multiplexer or selection step. The symbol 200 denotes a square root operator block.
3 The other symbols in FIG. 37 have meanings similar to the like symbols in FIG. 36.

4 To simplify audio buffer management during splicing transients, it is
5 recommended to have the same audio buffer levels at the beginning and at the end of the
6 clips. The case of going from a low to a high audio buffer level is the most problematic,
7 and is addressed by a sufficiently precise mean buffer level estimate for beyond the
8 selected In Point.

9 If there are multiple audio streams for one program, then all of these individual
10 audio streams are processed independently in the fashion described above for a single
11 stream. For example, there could be two stereo audio streams for one program, or four
12 audio streams for quadrasonic sound. The association of the ending (i.e., first) clip and
13 starting (i.e., second) clip audio streams to splice together depends on the PID of the
14 streams after PID re-mapping, if there is PID re-mapping, or on the PID of each stream in
15 the spliced clips, if there is no PID re-mapping. For an audio stream of the ending clip
16 that has no audio stream in the starting clip that can be associated with it, the preserved
17 audio packets are played until the end. This will achieve the best possible alignment
18 between audio and video for the ending clip.

19 The method used above for seamless audio splicing can also be used for splicing
20 other elementary streams containing encapsulated data. For example, a TS may have
21 additional elementary streams of other data encapsulated in access units such as access
22 units for teletext, closed captioning, VBI, etc. To apply the seamless splicing method to a
23 TS having multiple elementary streams of non-video and non-audio access units, the

1 AU's in each elementary stream are found that are best aligned with the first and last
 2 video frames, and an AU sequence over the splice is selected, independent of the content
 3 of the other non-video elementary streams. In this case, the method will minimize skew
 4 with respect to associated video frames and also prevent accumulation of skew from
 5 multiple splices in the TS.

6 With reference to FIG. 38, there is shown a flow chart of a procedure for
 7 calculating the video time stamp offset V_{OFFSET} . In a first step 211, the procedure finds
 8 the DTS of the last video frame (in decode order) of the first clip. This DTS of the last
 9 video frame of the first clip is denoted DTS_{VL1} . Then in step 212, the procedure finds the
 10 original DTS of the first frame to be decoded in the second clip. This DTS of the first
 11 frame to be decoded in the second clip is denoted DTS_{VF2} . Finally, in step 213, the video
 12 time stamp offset V_{OFFSET} is computed as $\text{DTS}_{\text{VL1}} - \text{DTS}_{\text{VF2}}$ plus one video frame duration.

13 With reference to FIG. 39, there is shown a flow chart of a procedure for
 14 calculating the audio time stamp offset A_{OFFSET} . In a first step 221, the procedure finds
 15 the PTS of the last AAU of the first clip. This PTS of the last AAU of the first clip is
 16 denoted PTS_{AL1} . Then in step 222, the procedure finds the original PTS of the first AAU
 17 to be decoded in the second clip. This PTS of the first AAU to be decoded in the second
 18 clip is denoted PTS_{AI2} . Finally, in step 223, the audio time stamp offset A_{OFFSET} is
 19 computed as $\text{PTS}_{\text{AL1}} - \text{PTS}_{\text{AI2}}$ plus one AAU duration.

20 With reference to FIG. 40, there is shown a flow chart of a procedure for
 21 calculating the PCR offset $\text{PCR}_{\text{OFFSET}}$. In a first step 231, the procedure finds the
 22 extrapolated PCR_e for the last byte of the first clip. This extrapolated PCR_e is denoted
 23 PCR_{eL1} . Then in step 232, the procedure finds the original extrapolated PCR_e for the first

1 of the scene which occurs some frames after the splicing point (sometimes 10 frames). In
2 other decoders the consequences could be more catastrophic.

3 To avoid any unpleasant effect in a controlled fashion, the frames which cannot
4 be decoded are replaced by encoded frozen frames. These frames are encoded such that
5 they effectively repeat a previous frame in the decoding order. They can be either B-
6 frames or P-frames. The frozen frame implementation relies on null motion vectors and
7 no coded transform coefficients. Consequently, these frames are completely MPEG-2
8 compliant and the decoder doesn't encounter any discontinuity in the stream.

9 With these frozen frames, decoder freeze can be controlled to make the visual
10 perception cleaner. There are three different types of encoded frozen frames that can be
11 generated for this purpose. These three types are a P-frame repeating the previous I or P
12 frame (in display order), a B-frame repeating the previous I or P frame (in display order),
13 and a B-frame repeating the following I or P frame (in display order). Moreover, any
14 frozen frame should not be separated from the frame it is repeating by some live (i.e. non-
15 frozen) frames in display order. To avoid any undesirable flickering effect due to the
16 presence of two fields within an interlaced frame, the frozen frames are generated using
17 the dual motion prediction type which allows the encoding of one field by extrapolation
18 (prediction) from the dual field.

19 With reference to FIG. 42, there is shown a diagram of the pixels in a video frame
20 250. According to the MPEG video encoding standard, the video frame can be
21 subdivided into a rectangular array of macroblocks, where each macroblock 251 includes
22 a square array of pixels. Pixels on the lower right and lower borders of a frame that do
23 not fit into full size macroblocks are handled as follows. The frame horizontal and

vertical sizes are completed to the nearest integer multiples of macroblock horizontal and vertical sizes by right-most column and lower-most row repetitions respectively. The MPEG standard also permits slices, or linear arrays of contiguous macroblocks, to be defined, with the maximum sized slices including an initial macroblock in a left-most column and a final macroblock in a right-most column. For example, a maximum size slice 255 is shown including all of the macroblocks in the third row of the macroblock matrix. A large number of consecutive macroblocks in a slice can be very efficiently encoded by a command to skip that number of macroblocks immediately after the initial macroblock in the slice. In case of a skip, the encoding information (i.e., the motion vectors and quantized DCT coefficients for the prediction error) is common to all skipped macroblocks and therefore is not repeated for each skipped macroblock.

It is possible to encode a "freeze frame" in various ways, such that the encoding of the "freeze frame" will result in a selected variable size. The smallest freeze frame will define the maximum number of skipped macroblocks and maximum size slices, and a null set of DCT coefficients for the prediction residual and zero valued displacement vectors. The largest freeze frame will define, for each of the non-skipped macroblocks, a set of zero valued DCT coefficients for the prediction residual and zero valued displacement vectors. Freeze frames of intermediate sizes can be defined by using different numbers of skipped macroblocks, and then various sizes of slices of macroblocks. Also, a slight adjustment can be made by padding. Padding is done by placing some stuffing bytes in the adaptation field (see FIG. 5).

With reference to FIG. 43, there is illustrated a problem of non-obsolete audio TS packets 260 that follow in the first clip after the end 261 of the video TS packet for the

1 beginning with the most advanced in time packets. Then, in step 273, the procedure
2 branches depending on whether or not “j” is greater than “k”. If “j” is not greater than
3 “k”, then all of the non-obsolete audio packets following the Out Point from the first TS
4 stream have been inserted into the second TS stream following the In Point so that they
5 no longer constitute a problem for the seamless video splicing. In this case, execution
6 branches to step 274 to change any remaining obsolete audio packets to null TS packets,
7 and the reformatting procedure of FIG. 44 is finished.

8 If “j” is greater than “k”, execution continues from step 273 to step 275. In step
9 275, for the remaining (j-k) non-obsolete audio packets from the first stream, the
10 procedure creates (j-k)*188 bytes of additional space for them in the spliced TS stream
11 prior to the video for the Out Point. This additional space must be generated so as to
12 maintain the $T_s = T_e + 8 / (\text{bit rate})$ condition of FIG. 24 for seamless video splicing. This
13 additional space can be entirely or partially provided by the space of the null TS packets
14 created in step 157, in which case these null TS packets are replaced with non-obsolete
15 audio packets. Any remaining ones of the non-obsolete audio packets are placed into the
16 space opened up by reducing the space taken by the video packets in the first stream prior
17 to the Out Point. After step 275, the re-formatting routine of FIG. 44 is finished.

18 The reformatting of the spliced TS stream after concatenation also includes steps
19 to ensure the continuity of associated individual streams across the splice point. The
20 same program specific information (PSI) tables must occur before and after the splicing
21 point. This is achieved by re-stamping all of the program identification indices (PIDs)
22 within the second clip with the associated stream PIDs of the first clip. The program
23 identification indices must be the same for the different component streams which form a

- 1 - the DTS-PCR_e at the beginning of this GOP (i.e. for its first frame) and
- 2 the ending delay $T_{\text{end}} = \text{DTS}_{L1} - T_e$ at the end of this GOP which can be
- 3 computed.
- 4 - the number of frames to play from this GOP which is not necessarily
- 5 equal to the full GOP size.

6

7 To rebuild this GOP, the procedure has access to the GOP structure and the size of each

8 frame. So, the last GOP is read in full into the memory. This is done only if the

9 procedure needs to terminate with an incomplete GOP. If a play-at-time interrupt arrives

10 during playing a clip, the procedure determines in advance the number of frames

11 remaining before the transition to the next clip to prepare the GOP.

12 The frames to be replaced by encoded frozen frames depend on the GOP

13 structure. This point will be illustrated by examples.

14

15 Example 1: Incomplete GOP with 3n frames.

16

17 Transport order:	I	B	B	P	B	B	P	B	B	P	B	B
18 Display order:	2	0	1	5	3	4	8	6	7	11	9	10

19

20 Case 1: The procedure has to play 3n frames. The procedure takes the first 3n

21 frames without any problem since the set of the first 3n frames in the transport order is

22 the same as the set of the first 3n frames in display order as shown above.

23

1 Display order: 0 3 1 2 6 4 5 9 7 8 12 10 11

2

3

4 Within this GOP structure playing $3n+1$ frames is trivial and can be achieved
5 without any freeze frames. Playing $3n+2$ frames can be achieved by freezing just one
6 frame as illustrated below for the case of $3n+2=8$:

7

8

9 Transport order: I P B B P B B Pff

10 Display order: 0 3 1 2 6 4 5 7

11

12 where Pff implements a freeze of P6. Similarly, playing $3n$ frames can be
13 achieved by freezing two frames as illustrated below for the case of $3n=9$:

14

15 Transport order: I P B B P B B Pff Bff

16 Display order: 0 3 1 2 6 4 5 8 7

17

18 where Pff and Bff both implement a freeze of P6.

19

20 These changes are applied before taking into account the buffer level. They provide a
21 modified GOP tailored for achieving the desired temporal frame accuracy. After these
22 transformations related to the GOP structure are performed, the buffer level (DTS-PCR)
23 at the end of this GOP is computed based on the resultant (i.e. modified) GOP structure.

1 If the new GOP's (i.e. the first GOP of the clip to start) buffer level is too high
2 and if there is no padding bandwidth available in the end of the first clip, then additional
3 frames are replaced by encoded frozen frames, starting from the last one in transport
4 order and proceeding one frame at a time (towards the beginning of the first clip) until the
5 GOP size becomes small enough.

6 These GOP transformations can be done in advance, as soon as the number of
7 frames to play in the current clip becomes known. This means that, if there is a play-at-
8 time command to start the next clip, then the timer must expire late enough to allow the
9 computation of frames remaining to play and also the preparation of the last GOP.

10 With reference to FIG. 45, it is possible to pre-compute metadata that can speed
11 up the process of seamless splicing. This is especially useful when the seamless splicing
12 must be done on the fly, during real-time delivery of a TS stream. For example, a stream
13 server of the video file server 20 of FIG. 1 performs metadata computation (281 in FIG.
14 45) when the file server records the MPEG TS stream in a MPEG file 282. As the MPEG
15 TS data 285 becomes recorded in the MPEG file 282, the metadata is recorded in a
16 header of the MPEG file. The header, for example, is a first megabyte of random-
17 accessible address space in the file. Preferably, the metadata includes some metadata 283
18 associated with the clip as a whole, and metadata 284 associated with the individual
19 GOPs. Preferably, the metadata 284 associated with the individual GOPs is stored in a
20 GOP index table.

21 The metadata 283 associated with the clip as a whole includes a program number,
22 the video frame rate, status of the clip, the number of GOPs in the clip, stream identifiers
23 for the various elementary streams in the TS, a byte index indicating a beginning position

1 2 over the GOP entry for GOP no. 1, writing the content of the GOP entry for GOP no. 4
2 over the GOP entry for GOP no. 2, writing the content of the GOP entry for GOP no. 6
3 over the entry for GOP no. 3, etc.

4 With reference to FIG. 47, there is shown a flow chart for GOP index decimation.
5 In a first step 331, before computing attributes for any GOP, a GOP decimation factor is
6 set to one in the metadata for the clip. (This decimation factor, for example, is used to
7 find a GOP table index for a given GOP number by dividing the given GOP number by
8 the decimation factor.) Computation of attribute values for the GOPs found in an
9 ingested TS and the writing of those attribute values to respective entries in the GOP
10 index continues in step 332 until the end of the GOP index is reached in step 333. Then
11 the procedure continues to step 334 where the GOP index is decimated by a factor of two.
12 Finally, the decimation factor is increased by a factor of two, and the procedure loops
13 back to step 332.

14 Some of the metadata is of high priority and some of the metadata is of lower
15 priority. In the absence of sufficient computational resources, the high priority metadata
16 can be pre-computed without pre-computing the lower priority metadata. For example,
17 the frame rate for the clip is a high priority item but the number of frames in the clip is a
18 low priority item. The frame number and the pointer to the corresponding MPEG TS
19 data (i.e., a byte index) are high priority GOP attributes. The flag indicating whether or
20 not the GOP is open or closed is a low priority item. In the situation where it is possible
21 that a GOP entry will include the high priority items but not the low priority items, the
22 low priority items are encoded with an indication of whether they are valid or not. This

1 can be done by initially setting the low priority items to predetermined invalid values
2 indicating that valid attribute values are not yet computed.

3 With reference to FIG. 48, there is shown a flow chart of metadata computations
4 for a next GOP processed in a TS. In a first step 341, if resources available for
5 computing high priority metadata are not presently available, then the computations for
6 the GOP are terminated. Otherwise, the procedure continues to step 342, where the high
7 priority metadata is computed for the GOP. Then, in step 343, if resources for computing
8 low priority metadata are not available, then the computations for the GOP are
9 terminated. Otherwise, the procedure continues to step 344, where the low priority
10 metadata is computed for the GOP.

11 The GOPs in a TS can be fixed size (same size throughout the TS) or variable size
12 in terms of the number of video frames they contain. If the GOPs are of a fixed size, then
13 each has an integral number of "n" frames. In this case, assuming that the first frame
14 number in the TS is "m", then the number of the GOP containing a specified frame "p"
15 can be calculated as the integer quotient of (p-m)/n plus one. If the GOPs are of variable
16 size, then the metadata may include an average GOP size; i.e., an average number of
17 frames per GOP. In this case, to find the GOP containing a specified frame, the GOP
18 number is estimated using the same formula (using the average number of frames per
19 GOP for n), and then the GOP index table is searched in the neighborhood of this GOP
20 for the GOP containing the specified frame number.

21 The metadata contains information on the clip which is used during the play
22 operation to check the buffer levels and to adjust these levels at the splicing time. The
23 fundamental information item of metadata is the difference $DTS-PCR_c$ for each video

1 access unit within the video stream which is representative of the buffer level in the sense
 2 described previously. It should be noted that DTS values are defined for I and P frames
 3 for which the decoding and presentation times differ since these frames are used as
 4 references by other P and B frames. However, for type B frames only PTS is defined
 5 which is identical to the DTS of the same frame.

6 A subsection of the metadata includes the following two values:

7
 8 First PTS: This is the PTS of the first frame in display order.

9
 10 First PCR, ($PCR_{e,0}$): This is a calculated (i.e., extrapolated) PCR value corresponding
 11 to the beginning (i.e. the first byte) of the file. This value is computed from the bit-rate,
 12 the value of the first genuine PCR record and its byte position within the file.

13
 14 Based on these two values, for each I frame the procedure computes both the DTS
 15 of this frame and also the PCR_e value corresponding to the beginning of this frame within
 16 the file. In order to perform these calculations, the procedure also accesses the frame
 17 number (a cumulative frame count from the beginning of the file) and the byte position of
 18 the beginning of this frame in the file, both of which are recorded in the GOP index table.

19 The GOP index table forms a major sub-section of the metadata. It is easy to see
 20 that assuming one I frame per GOP, the cumulative frame count values at I pictures also
 21 become their cumulative temporal references (referring to display order). Then, it is
 22 straightforward to calculate a PTS value for each of these I frames assuming a continuous
 23 video play-out. Finally, assuming a known uniform GOP structure, these presentation

1 time stamps of I pictures can be easily converted to decoding time stamps based on the
2 principle that the decode time instant of an anchor frame is the same as the presentation
3 time instant of the previous anchor frame. So, the DTS-PCR_e difference can be
4 computed in advance for each I frame of the file and consequently whatever the start
5 position is in a clip for play-out, the required buffer level to be build-up can be known in
6 advance.

7 With reference to FIG. 49, there are shown further details of the components
8 involved in the ingestion of an MPEG TS into a stream server computer 291 for
9 recording in the cached disk array, and for real-time splicing during real-time
10 transmission of an MPEG TS from the cached disk array and from the stream server
11 computer to a destination such as one of the clients (54 in FIG. 1). The stream server
12 computer 291 is interfaced to the network (25 in FIG. 1) via a network interface board
13 292. The network interface board 292, for example, is a DVB board, an ATM board, an
14 Ethernet board, a Fiber Channel board, or a Gigabit Ethernet board. The network
15 interface board 292 performs a direct memory access upon buffers 293 in the random
16 access memory 294 of the stream server computer 291 in order to exchange MPEG TS
17 data with the network (25 in FIG. 1). A software driver 295 for the network interface
18 board 292 initiates the direct memory access transfers. In particular, the software driver
19 295 hands to the network interface board 292 the RAM address range of the data in the
20 buffer for the DMA transfer. Real-time delivery of an MPEG TS stream from the stream
21 server 291 is controlled by a "house" clock signal 55. As shown in FIG. 1, the house
22 clock signal 55 is applied to each of the stream servers 21 and the controller servers 28,

1 29 in the video file server 20. This house clock signal 55 simultaneously interrupts each
2 stream server and controller server at the video frame rate.

3 For DVB (digital video broadcast), data is transmitted upon request. When the
4 stream server is accepting data from an application, the request is produced when a
5 receive buffer becomes available. For ATM (asynchronous transfer mode), the data is
6 transmitted in response to a time interrupt signal. A buffer is scheduled to be available
7 when the interrupt is expected to occur. In either case, when transmitting an MPEG TS,
8 the data must be delivered to ensure that any jitter is within the limit that the MPEG
9 standard imposes on the PCR time values. The PCR values must be accurate within 20
10 cycles of a 27 MHz decoder clock. Moreover, the difference between neighboring PCR
11 values in the TS is kept less than 100 msec; otherwise, the decoder clock will reset.

12 When ingesting an MPEG TS from the network (25 in FIG. 1), once an assigned
13 one of the buffers 293 is filled with MPEG TS data, the software driver 295 inserts a
14 pointer to the filled buffer into a FIFO buffer pointer queue 296. A metadata
15 computation software program module 297 finds that the queue 296 is not empty, and
16 services the queue by obtaining the buffer pointer from the queue and accessing the
17 MPEG TS data in the buffer 293 indicated by the pointer. The metadata computed by the
18 program module 297, for example, is placed in a header of the buffer. When the
19 metadata computation module 297 is finished, it places the buffer pointer in another FIFO
20 buffer pointer queue 298. The queue 298 is serviced by a write access program module
21 299. The write access program module 299 removes the pointer from the queue 298, and
22 then writes the data from the indicated buffer to an MPEG TS file of the file system 300.
23 The file system 300 writes the data to the cached disk array 23 in an asynchronous write

1 per second. The buffers smooth out variations in bit rate that are not deterministic in
2 time. The buffer size can be much smaller than a clip, and smaller than a portion of a clip
3 that is needed for splicing. In this case, when splicing a first stream (S_1) to a second
4 stream (S_2), alternate buffers in sequence around the carousel can contain data from the
5 same clip. For example, a first buffer may contain a next-to-last segment of the first clip,
6 a second buffer may contain a first segment of the second clip, a third buffer may contain
7 a last segment of the first clip, and a fourth buffer may a second segment of the second
8 clip.

9 The metadata computation module 297 parses the content of its assigned buffer.
10 The parsing typically continues from one buffer to a next buffer in the carousel, for the
11 usual case where each buffer has a size smaller than the duration of the TS. The parsing
12 counts frames, builds GOP entries, calculates instantaneous bit rates and other GOP
13 attributes, and looks for error conditions. Each GOP header is parsed for display order
14 and type (i.e., open or closed).

15 The MPEG processing 303 may use a number of flags. These MPEG processing
16 flags include the following:

17

18 - 0x100: re-stamp time records flag

19

20 If this flag is set then all of the PCR and DTS/PTS records are recomputed and re-
21 stamped so that they are continuous across splicing transitions.

22

23 - 0x200 : discontinuity flag

SECRET

4

6

12

14

18

20

1 With reference to FIG. 50, there is shown a diagram that illustrates a metered file
2 transfer protocol (FTP). This protocol is useful for transfer of an MPEG TS stream from
3 the video file server (20 in FIG. 1) to an application 310. The application 310, for
4 example, is a client application on the network 25, or it could be another video file server.
5 The application 310 initiates the metered FTP by sending a copy command to the active
6 one of the controller servers 28, 29. The controller server sends a set bandwidth
7 command to the stream server to set the bit rate for the metered transfer of file data
8 between the stream server 291 and the application 310. The stream server then issues a
9 connect message to the application to open an IP channel for the transfer of the file data.
10 In the metered FTP protocol, the data transmission rate is controlled so that the loading
11 on the stream server is deterministic. The data transmission is TCP flow controlled. For
12 input to the stream server from the application, the stream server controls the data rate by
13 flow-control push-back. For transmission of data from the stream server to the
14 application, the stream server merely controls the rate at which it transmits the data.

15 In the transmission control protocol (TCP), the stream server either opens or
16 closes a window of time within which to receive more data. The stream server indicates
17 to the application a certain number of buffers that are available to receive the data. In
18 addition, the stream sever acknowledges receipt of the data.

19 In the metered FTP protocol, time is split up into one-second intervals, and at
20 every 1/10 of a second, the average data rate is re-computed. An adjustment is made to a
21 data transmission interval parameter if the computed data rate deviates from the desired
22 rate. For example, for a desired 10 kilobyte per second transmission rate, the data
23 transmission size is set at one kilobyte, and the data transmission interval parameter is

09540305 "03:31:00

1 initially set at 1/10 of a second. If the computed average data rate happens to be less than
2 10 kilobytes per second, then a one kilobyte bunch of data will be transmitted more
3 frequently than once every 1/10 of a second.

4 With reference to FIG. 51, there is shown a control station play list 320 and a
5 stream server play list 321. As described in the above referenced Duso et al. U.S. Patent
6 5,892,915, the stream server can continuously supply a real-time data stream to a client
7 by servicing a play list. As shown in FIG. 51, the play list can be distributed between the
8 controller server play list 320 and a stream server play list 321. The controller server
9 play list, for example, holds up to 356 elements that point to clips, and the stream server
10 play list holds up to three of the elements. The stream server play list is in effect a
11 window to up to three of the elements that are at or near the head of the controller server
12 play list. When a pointer to a clip is placed in a play-list entry, the entry also has an
13 indication of an In Point in the clip, an Out Point for the clip, and a desired start time. A
14 pointer to a clip can be recalled (i.e., removed) from a play-list entry.

15 When a pointer to a clip is appended to the controller server play list 320, extra
16 audio packets are loaded in a FIFO buffer. Then, the start and the end positions of the
17 clip are set based on the video elementary stream only. The clip is read starting with the
18 first video frame to play and until the end of the last frame to play. One dedicated buffer
19 is associated with each audio stream. The number of additional audio access units to play
20 is computed at the splicing time. All of these pre-fetched audio access units will be
21 played only if the clip is played until the end. However, if the play-out of the clip is
22 interrupted by a "play-next immediate" or a "play-next at-time" command then some of

1 the preloaded extra audio is replaced by audio data (i.e. audio access units) extracted
2 from the new clip's buffer pool at the splicing time.

3 The seamless splicing techniques described above can also be used to recover
4 from failure conditions that may destroy or corrupt a portion of an MPEG transport
5 stream. For example, a component of a data path in the cached disk array may fail,
6 causing an MPEG TS from a disk drive in the cached disk array to be interrupted for a
7 short period of time while the failure condition is diagnosed and the MPEG TS is re-
8 routed to bypass the failed component. As shown in the flow chart of FIG. 52, the MPEG
9 processing module may be programmed to recognize the failure (step 351) during the
10 delivery of the MPEG TS to a client (step 352). Once this failure is detected, the MPEG
11 processing module 303 can fill in this gap in the MPEG TS with null packets or freeze
12 frames with correct PCR values (step 353). By inserting correct PCR values at less than
13 the required minimum interval (less than 100 milliseconds), a client's decoder will not
14 reset and can be kept in a ready state. Once delivery of the MPEG TS to the MPEG
15 processing module is reestablished (as detected in step 354), the MPEG processing
16 module seamlessly splices (step 355) the re-established TS (as if it were a second stream
17 or clip) to the TS of null packets or freeze frames that it has been generating and sending
18 to the client. Splicing could be performed in a similar fashion in the set-top decoder box
19 of FIG. 2 or the switch of FIG. 3 to compensate for temporary interruptions in the
20 delivery of an MPEG TS to the set-top decoder box or to the switch.

21 In a similar fashion, the MPEG processing module in batch mode could check a
22 clip for any damaged portions, and once a damaged portion is found, remove it by
23 seamlessly splicing the end of the first good part of the clip to the beginning of the last

1 good part of the clip. Batch mode processing also would have the advantage that the
2 audio and video buffer levels could be determined exactly by simulation, so that it would
3 be possible to guarantee the absence of any buffer underflow or overflow at every point
4 after the splice. Batch mode processing, with audio and video buffer simulators, could
5 also measure the quality of spliced TS streams and determine whether or not the splices
6 should be repaired using the more accurate simulated buffer levels. The quality
7 measurement could also include an analysis of audio delay or skew; how many freeze
8 frames are in the TS stream and their clustering, and an analysis of PCR jitter. It would
9 also be very easy for the MPEG processing module to compute the audio skew and PCR
10 jitter in real time during the real-time transmission of an MPEG TS, and to display
11 continuous traces of the audio skew and PCR jitter to a system administrator.

12 In view of the above, there has been described the preparation of metadata for
13 splicing of an encoded digital motion video stream (such as an MPEG Transport Stream)
14 is prepared in real time while recording at the encoding bit rate and faster than encoded
15 bit rate for off line encoding independent of the bit rate and mechanisms for ingestion of
16 the data stream into data storage. Preprocessing is performed during a metered file
17 transfer protocol (FTP) and includes pseudo real-time encoding. The preprocessing
18 includes Group of Pictures (GOP) level pre-processing of splicing In Points and results in
19 an intimate linkage between metadata and the file system in which the video data is
20 stored. The preferred file system enables access to metadata in parallel to writing the
21 data on disk. The pre-processing is performed simultaneous to writing the data to the
22 disk using a carousel type buffer mechanism.